# EXHIBIT M

**U.S. Patent No. 9,923,995 – Google LLC**
**Claims 28 and 29**

Jenam Tech LLC ("Jenam") provides evidence of infringement of claims 28 and 29 of U.S. Patent No. 9,923,995 (hereinafter "the '995 patent") by Google LLC ("Google"). In support thereof, Jenam provides the following claim charts.

"Accused Instrumentalities" as used herein refers to at least one or more websites or web addresses including, but not limited to www.google.com, stored and/or hosted on one or more servers owned or under the control of Google. These claim charts demonstrate Google's infringement, and provide notice of such infringement, by comparing each element of the asserted claims to corresponding components, aspects, and/or features of the Accused Instrumentalities. These claim charts are not intended to constitute an expert report on infringement. These claim charts include information provided by way of example, and not by way of limitation.

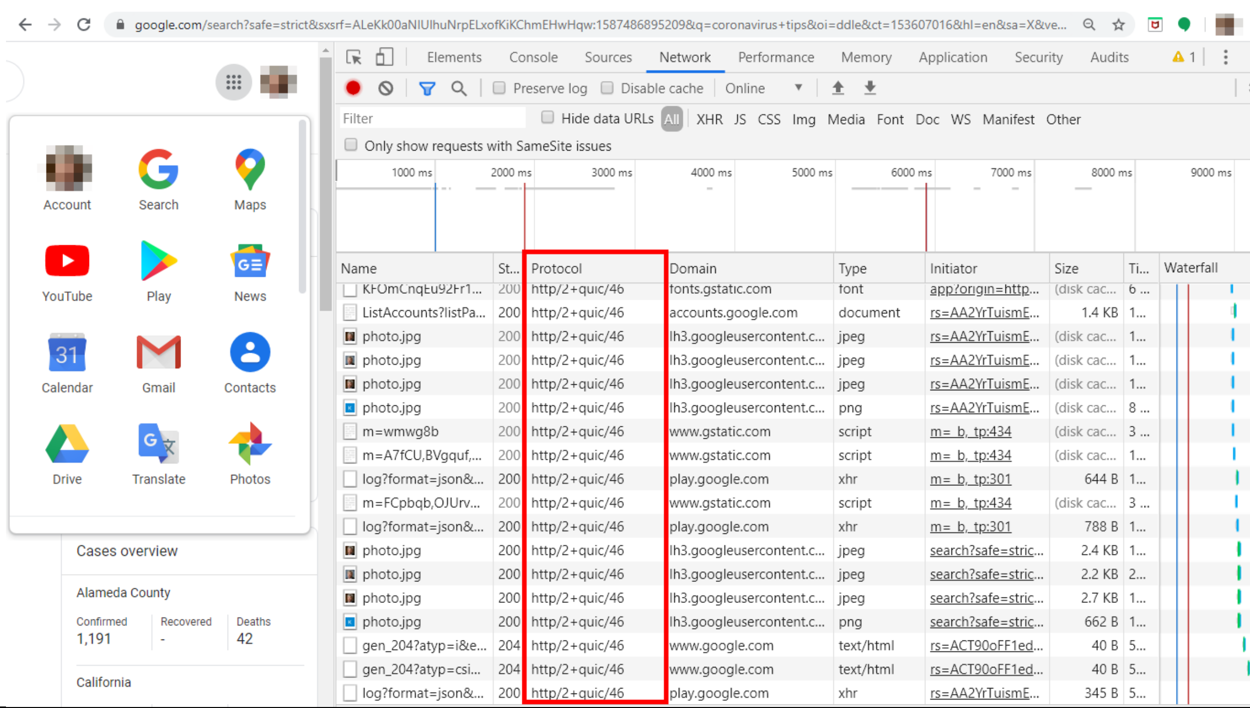
The analysis set forth below is based only upon information from publicly available resources regarding the Infringing Instrumentalities, as Google has not yet provided any non-public information. An analysis of Google's (or other third parties') technical documentation and/or software source code may assist in fully identify all infringing features and functionality. Accordingly, Jenam reserves the right to supplement this infringement analysis once such information is made available to Jenam. Furthermore, Jenam reserves the right to revise this infringement analysis, as appropriate, upon issuance of a court order construing any terms recited in the asserted claims.

Unless otherwise noted, Jenam contends that Google directly infringes the '995 patent in violation of 35 U.S.C. § 271(a) by selling, offering to sell, making, using, and/or importing the Infringing Instrumentalities. The following exemplary analysis demonstrates that infringement. Unless otherwise noted, Jenam further contends that the evidence below supports a finding of indirect infringement under 35 U.S.C. §§ 271(b) and/or (c), in conjunction with other evidence of liability under one or more of those subsections. Google makes, uses, sells, imports, or offers for sale in the United States, or has made, used, sold, imported, or offered for sale in the past, without authority, or induces others to make, use, sell, import, or offer for sale in the United States, or has induced others to make, use, sell, import, or offer for sale in the past, without authority products, equipment, or services that infringe claims 28 and 29 of the '995 patent, including without limitation, the Accused Instrumentalities.

Unless otherwise noted, Jenam believes and contends that each element of each claim asserted herein is literally met through Google's provision of the Infringing Instrumentalities. However, to the extent that Google attempts to allege that any asserted claim element is not literally met, Jenam believes and contends that such elements are met under the doctrine of equivalents. More specifically, in its investigation and analysis of the Infringing Instrumentalities, Jenam did not identify any substantial differences between the elements of the patent claims and the corresponding features of the Infringing Instrumentalities, as set forth herein. In each instance, the identified feature of the Infringing Instrumentalities performs at least substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim element.

To the extent the chart of an asserted claim relies on evidence about certain specifically-identified Accused Instrumentalities, Jenam asserts that, on information and belief, any similarly-functioning instrumentalities also infringes the charted claim. Jenam reserves the right to amend this infringement analysis based on other products made, used, sold, imported, or offered for sale by Google. Jenam also reserves the right to amend this infringement analysis by citing other claims of the '995 patent, not listed in the claim chart, that are infringed by the Accused Instrumentalities. Jenam further reserves the right to amend this infringement analysis by adding, subtracting, or otherwise modifying content in the "Accused Instrumentalities" column of each chart.

CLAIM CHARTS
BASED ON INFRINGEMENT ANALYSIS OF GOOGLE
U.S. Patent No. 9,923,995

| Claim 28 | Accused Instrumentalities |
|---|---|
| An apparatus comprising:<br><br>a non-transitory memory storing a network application; and<br><br>one or more processors in communication with the non-transitory memory, wherein the one or more processors execute the network application such that the network application is configured to operate in accordance with a non-transmission control protocol (TCP) protocol that operates above an Internet Protocol (IP) layer and below a hypertext transfer protocol (HTTP) application layer, the apparatus, when operating in accordance with the non-TCP protocol, configured to: | Google uses an apparatus (e.g., one or more servers, etc.) including a non-transitory memory storing a network application, and one or more processors in communication with the non-transitory memory, wherein the one or more processors in communication with the non-transitory memory, wherein the one or more processors execute the network application (e.g., server software, etc.) such that the network application is configured to operate in accordance with a non-transmission control protocol (TCP) protocol (e.g., the QUIC protocol, etc.) that operates above an Internet Protocol (IP) layer and below a hypertext transfer protocol (HTTP) application layer.<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**Note**: Below is a web page of Google (https://www.google.com/).<br><br> |

| | |
|---|---|
| receive idle information for use in detecting an idle time period that results in a non-TCP connection being subject to deactivation; | Google uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, mandates receiving idle information for use in detecting an idle time period (e.g., idle timeout parameter field, etc.) that results in a non-TCP connection being subject to deactivation;<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**<u>Note</u>**: As set forth below, a QUIC negotiation packet includes transport parameters that include an idle timeout parameter that is detected by a recipient of such packet.<br><br>7.4. Transport Parameters<br><br>During connection establishment, both endpoints make authenticated declarations of their transport parameters. These declarations are made unilaterally by each endpoint. Endpoints are required to comply with the restrictions implied by these parameters; the description of each parameter includes rules for its handling. |

| | |
|---|---|
| | <table><tr><th>Value</th><th>Parameter Name</th><th>Specification</th></tr><tr><td>0x0000</td><td>initial_max_stream_data</td><td>Section 7.4.1</td></tr><tr><td>0x0001</td><td>initial_max_data</td><td>Section 7.4.1</td></tr><tr><td>0x0002</td><td>initial_max_stream_id</td><td>Section 7.4.1</td></tr><tr><td>0x0003</td><td>idle_timeout</td><td>Section 7.4.1</td></tr><tr><td>0x0004</td><td>omit_connection_id</td><td>Section 7.4.1</td></tr><tr><td>0x0005</td><td>max_packet_size</td><td>Section 7.4.1</td></tr><tr><td>0x0006</td><td>stateless_reset_token</td><td>Section 7.4.1</td></tr></table> Table 4: Initial QUIC Transport Parameters Entries<br>The format of the transport parameters is the TransportParameters struct from Figure 6. This is described using the presentation language from Section 3 of [I-D.ietf-tls-tls13]. |

<table>
<tr>
<td></td>
<td><pre>uint32 QuicVersion;

enum {
   initial_max_stream_data(0),
   initial_max_data(1),
   initial_max_stream_id(2),
   idle_timeout(3),
   omit_connection_id(4),
   max_packet_size(5),
   stateless_reset_token(6),
   (65535)
} TransportParameterId;</pre></td>
</tr>
<tr>
<td>generate, based on the idle information, a non-TCP packet including an idle time period parameter field identifying metadata that is specified in at least one of a number of seconds or minutes; and</td>
<td>Google uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, mandates generating, based on the idle information, a non-TCP packet (e.g., QUIC negotiation packet, etc.) including an idle time period parameter field (e.g., idle timeout parameter field, etc.) identifying metadata (e.g., the value of the idle timeout parameter field, etc.) that is specified in at least one of a number of seconds or minutes; and<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br><u>Note</u>: As set forth below, since the idle_timeout value sets the duration of idleness, after which the connection is shutdown, a timeout attribute of the connection is necessarily modified based on the value received in the idle_timeout field of the connection negotiation packet.<br><br><pre>idle_timeout (0x0003):  The idle timeout is a value in seconds that
   is encoded as an unsigned 16-bit integer.  The maximum value is
   600 seconds (10 minutes).</pre></td>
</tr>
</table>

| | |
|---|---|
| | 7.8. Connection Termination<br><br>Connections should remain open until they become idle for a pre-negotiated period of time. A QUIC connection, once established, can be terminated in one of three ways:<br><br>7.8.2. Idle Timeout<br><br>A connection that remains idle for longer than the idle timeout (see Section 7.4.1) becomes closed. Either peer removes connection state if they have neither sent nor received a packet for this time.<br><br>The time at which an idle timeout takes effect won't be perfectly synchronized on peers. A connection enters the draining period when the idle timeout expires. During this time, an endpoint that receives new packets MAY choose to restore the connection. Alternatively, an endpoint that receives packets MAY signal the timeout using an immediate close. |
| send, from a first node to a second node and for establishing the non-TCP connection, the non-TCP packet to provide the metadata to the second node, for use by the second node in determining a timeout attribute associated with the non-TCP connection; | Google uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, mandates sending, from a first node to a second node and for establishing the non-TCP connection, the non-TCP packet (e.g., QUIC negotiation packet, etc.) to provide the metadata (e.g., the value of the idle timeout parameter field, etc.) to the second node, for use by the second node in determining a timeout attribute associated with the non-TCP connection;<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**<u>Note</u>**: As set forth below, a QUIC negotiation packet includes transport parameters that include an idle timeout parameter that is detected by a recipient of such packet. |

| | |
|---|---|
| | 7.4. Transport Parameters<br><br>During connection establishment, both endpoints make authenticated declarations of their transport parameters. These declarations are made unilaterally by each endpoint. Endpoints are required to comply with the restrictions implied by these parameters; the description of each parameter includes rules for its handling.<br><br>Value \| Parameter Name \| Specification<br>0x0000 \| initial_max_stream_data \| Section 7.4.1<br>0x0001 \| initial_max_data \| Section 7.4.1<br>0x0002 \| initial_max_stream_id \| Section 7.4.1<br>0x0003 \| idle_timeout \| Section 7.4.1<br>0x0004 \| omit_connection_id \| Section 7.4.1<br>0x0005 \| max_packet_size \| Section 7.4.1<br>0x0006 \| stateless_reset_token \| Section 7.4.1<br><br>Table 4: Initial QUIC Transport Parameters Entries |

| | |
|---|---|
| | The format of the transport parameters is the TransportParameters struct from Figure 6. This is described using the presentation language from Section 3 of [I-D.ietf-tls-tls13].<br><br>`uint32 QuicVersion;`<br><br>`enum {`<br>`initial_max_stream_data(0),`<br>`initial_max_data(1),`<br>`initial_max_stream_id(2),`<br>`idle_timeout(3),`<br>`omit_connection_id(4),`<br>`max_packet_size(5),`<br>`stateless_reset_token(6),`<br>`(65535)`<br>`} TransportParameterId;`<br><br>**Note**: As set forth below, since the idle_timeout value sets the duration of idleness, after which the connection is shutdown, a timeout attribute of the connection is necessarily modified based on the value received in the idle_timeout field of the connection negotiation packet.<br><br>idle_timeout (0x0003): The idle timeout is a value in seconds that is encoded as an unsigned 16-bit integer. The maximum value is 600 seconds (10 minutes).<br><br>7.8. Connection Termination<br><br>Connections should remain open until they become idle for a pre-negotiated period of time. A QUIC connection, once established, can be terminated in one of three ways: |

| | |
|---|---|
| | 7.8.2. Idle Timeout<br><br>A connection that remains idle for longer than the idle timeout (see Section 7.4.1) becomes closed. Either peer removes connection state if they have neither sent nor received a packet for this time.<br><br>The time at which an idle timeout takes effect won't be perfectly synchronized on peers. A connection enters the draining period when the idle timeout expires. During this time, an endpoint that receives new packets MAY choose to restore the connection. Alternatively, an endpoint that receives packets MAY signal the timeout using an immediate close. |
| wherein the apparatus, when operating in accordance with the TCP protocol, is configured to perform a three-way TCP handshake for establishing a TCP connection that is separate from the non-TCP connection. | Google uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, is configured wherein the apparatus, when operating in accordance with the TCP protocol, is configured to perform a three-way TCP handshake for establishing a TCP connection that is separate from the non-TCP connection.<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>"The synchronization requires each side to send its own initial sequence number and to receive a confirmation of it in acknowledgment from the other side. Each side must also receive the other side's initial sequence number and send a confirming acknowledgment.<br><br>1) A --> B SYN my sequence number is X<br>2) A <-- B ACK your sequence number is X<br>3) A <-- B SYN my sequence number is Y<br>4) A --> B ACK your sequence number is Y<br><br>Because steps 2 and 3 can be combined in a single message <u>this is called the three way (or three message) handshake</u>. |

| | |
|---|---|
| | A three way handshake is necessary because sequence numbers are not tied to a global clock in the network, and TCPs may have different mechanisms for picking the ISN's. The receiver of the first SYN has no way of knowing whether the segment was an old delayed one or not, unless it remembers the last sequence number used on the connection (which is not always possible), and so it must ask the sender to verify this SYN. The three way handshake and the advantages of a clock-driven scheme are discussed in [3]."<br><br>"Request for Comments" (RFC) document RFC 793 (September 1981) https://tools.ietf.org/html/rfc793 |

| Claim 29 | Accused Instrumentalities |
| --- | --- |
| The apparatus of claim 28 wherein the apparatus is configured such that:<br><br>the determination of the timeout attribute results from a negotiation between the first node and the second node; | Google infringes claim 28 and uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, mandates that the determination of the timeout attribute results from a negotiation between the first node and the second node;.<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**Note**: As set forth below, a QUIC negotiation packet is received by the client node from a server node.<br><br>7.4. Transport Parameters<br><br>During connection establishment, both endpoints make authenticated declarations of their transport parameters. These declarations are made unilaterally by each endpoint. Endpoints are required to comply with the restrictions implied by these parameters; the description of each parameter includes rules for its handling.<br><br>QUIC packet: A well-formed UDP payload that can be parsed by a QUIC receiver. QUIC packet size in this document refers to the UDP<br><br>**Note**: As set forth below, prior to a QUIC connection being established, the QUIC connection is "set up" using the aforementioned handshake. |

| | |
|---|---|
| | ```<br>3.1.  Low-Latency Connection Establishment<br><br>   QUIC relies on a combined cryptographic and transport handshake for<br>   setting up a secure transport connection.  QUIC connections are<br>   expected to commonly use 0-RTT handshakes, meaning that for most QUIC<br>   connections, data can be sent immediately following the client<br>   handshake packet, without waiting for a reply from the server.  QUIC<br>   provides a dedicated stream (Stream ID 0) to be used for performing<br>   the cryptographic handshake and QUIC options negotiation.  The format<br>   of the QUIC options and parameters used during negotiation are<br>   described in this document, but the handshake protocol that runs on<br>   Stream ID 0 is described in the accompanying cryptographic handshake<br>   draft [QUIC-TLS].<br>``` |
| during the idle time period, no non-TCP packet including data is communicated in the non-TCP connection; | Google infringes claim 28 and uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, mandates that, during the idle time period, no non-TCP packet including data is communicated in the non-TCP connection;<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**Note**: As set forth below, the metadata includes a value in seconds that is encoded as an unsigned 16-bit integer.<br><br>idle_timeout (0x0003):  The idle timeout is a value in seconds that<br>   is encoded as an unsigned 16-bit integer.  The maximum value is<br>   600 seconds (10 minutes).<br><br>**Note**: As set forth below, there is "no activity" while the connection is idle. |

| | |
|---|---|
| | 7.8.2. Idle Timeout<br><br>A connection that remains idle for longer than the idle timeout (see Section 7.4.1) becomes closed. Either peer removes connection state if they have neither sent nor received a packet for this time.<br><br>The time at which an idle timeout takes effect won't be perfectly synchronized on peers. A connection enters the draining period when the idle timeout expires. During this time, an endpoint that receives new packets MAY choose to restore the connection. Alternatively, an endpoint that receives packets MAY signal the timeout using an immediate close. |
| wherein the apparatus is further configured for:<br><br>detecting the idle time period based on the idle information; and | Google infringes claim 28 and uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, is configured wherein the apparatus is further configured for: detecting the idle time period (e.g., idle timeout parameter field, etc.) based on the idle information; and<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**<u>Note</u>**: As set forth below, a QUIC negotiation packet includes transport parameters that include an idle timeout parameter that is detected by a recipient of such packet.<br><br>7.4. Transport Parameters<br><br>During connection establishment, both endpoints make authenticated declarations of their transport parameters. These declarations are made unilaterally by each endpoint. Endpoints are required to comply with the restrictions implied by these parameters; the description of each parameter includes rules for its handling. |

| Value | Parameter Name | Specification |
|---|---|---|
| 0x0000 | initial_max_stream_data | Section 7.4.1 |
| 0x0001 | initial_max_data | Section 7.4.1 |
| 0x0002 | initial_max_stream_id | Section 7.4.1 |
| 0x0003 | idle_timeout | Section 7.4.1 |
| 0x0004 | omit_connection_id | Section 7.4.1 |
| 0x0005 | max_packet_size | Section 7.4.1 |
| 0x0006 | stateless_reset_token | Section 7.4.1 |

Table 4: Initial QUIC Transport Parameters Entries

The format of the transport parameters is the TransportParameters struct from Figure 6. This is described using the presentation language from Section 3 of [I-D.ietf-tls-tls13].

| | |
|---|---|
| | ```<br>uint32 QuicVersion;<br><br>enum {<br>   initial_max_stream_data(0),<br>   initial_max_data(1),<br>   initial_max_stream_id(2),<br>   idle_timeout(3),<br>   omit_connection_id(4),<br>   max_packet_size(5),<br>   stateless_reset_token(6),<br>   (65535)<br>} TransportParameterId;<br>``` |
| in response to detecting the idle time period, deactivating the non-TCP connection by at least partially closing the TCP-variant connection by one of the first and second nodes without communication between the second node and the first node that is related to the detection of the idle time period. | Google infringes claim 28 and uses the apparatus (e.g., the one or more servers, etc.) with the one or more processors that execute the instructions for communicating using the QUIC protocol which, in turn, is configured wherein, in response to detecting the idle time period, deactivating the non-TCP connection (e.g., connection termination, etc.) by at least partially closing the TCP-variant connection by one of the first and second nodes without communication between the second node and the first node that is related to the detection of the idle time period.<br><br>See excerpt(s) below, for example (emphasis added, if any):<br><br>**<u>Note</u>**: As set forth below, since the idle_timeout value sets the <u>duration of idleness, after which the connection is shutdown</u>, a timeout attribute of the connection is necessarily modified based on the value received in the idle_timeout field of the connection negotiation packet.<br><br>```<br>idle_timeout (0x0003):  The idle timeout is a value in seconds that<br>   is encoded as an unsigned 16-bit integer.  The maximum value is<br>   600 seconds (10 minutes).<br>``` |

| | |
|---|---|
| | 7.8. Connection Termination<br><br>Connections should remain open until they become idle for a pre-negotiated period of time. A QUIC connection, once established, can be terminated in one of three ways:<br><br>7.8.2. Idle Timeout<br><br>A connection that remains idle for longer than the idle timeout (see Section 7.4.1) becomes closed. Either peer removes connection state if they have neither sent nor received a packet for this time.<br><br>The time at which an idle timeout takes effect won't be perfectly synchronized on peers. A connection enters the draining period when the idle timeout expires. During this time, an endpoint that receives new packets MAY choose to restore the connection. Alternatively, an endpoint that receives packets MAY signal the timeout using an immediate close. |

**Caveat**: The notes and/or cited excerpts utilized herein are set forth for illustrative purposes only and are not meant to be limiting in any manner. For example, the notes and/or cited excerpts, may or may not be supplemented or substituted with different excerpt(s) of the relevant reference(s), as appropriate. Further, to the extent any error(s) and/or omission(s) exist herein, all rights are reserved to correct the same in connection with any subsequent correlations.